Background

Security Principal: **Securable objects** are AD entries that can have a Security Descriptor, such as: `Users`, `Groups` and `Computers`.
Security Descriptor: A list of all Access Control Entries of that object. The Security Descriptor is just an LDAP attribute. **Access Control Entry**: A specification of WHO can do WHAT to a specific Securable Object.
These permissions defined by the ACE can be seen if you go to any Securable Object (User, Group, or Computer) in `Active Directory Users and Computers`, such as the group "Domain Admins", or any user, and right-click and go to "Properties", and the click on "Security" and then on "Permissions".

As mentioned above, the ACE is specified in the attribute `nTSecurityDescriptor`, which is the type `NTSecurityDescriptor`. However, reading the value from that attribute is no easy, ADExplorer parses it like shown below. The only thing that makes any sense here are the SIDs. This is how ADExplorer shows an ACE:

```
D:AI(OA;;RP;4c164200-20c0-11d0-a768-00aa006e0529;;RS)(OA;;RP;5f202010-79a5-11d0-9020-00c04fc2d4cf;;RS)(OA;;RP;bc0ac240-79a9-11d0-902
```

There are two types of Access Control Lists:
Discrectionary Access Control List (DACL)
System Access Control List (SACL)

The interface for delegating control (creating ACE) in AD is not very clear. Delegation of control is usually performed by right-clicking on a OU and the click on **Delegate Control**, and then follow the wizard and add users/group/computer and specify which type of permission/control that securable object should have. In order to view the changes you first have to go to `ADUC/View/Advanced Features`, after that you will be able to right-click on a OU, select properties and then look at the `Security` tab, to see you delegated control changes. However, if you click on the actual securable object, such as a user, you will not see your changes, only if you click on `TheUser/Security/Advanced/Effective Access/SelectUser`.

As you can see, it is easy to delegate control and then forget about it.

What we are looking for is **security principals** that have some kind of interesting ACE on a high priviliged securable object. There are a ton of different ACEs, but only a handful are interesting from an attackers perspective. An ACE that allows the attacking user to compromise the victim user are what we are looking for. The following ACEs can do that:

```
 GenericAll - full rights to the object (add users to a group or reset user's password)
GenericWrite - update object's attributes (i.e logon script)
WriteOwner - change object owner to attacker controlled user take over the object
WriteDACL - modify object's ACEs and give attacker full control right over the object
AllExtendedRights - ability to add user to a group or reset password
ForceChangePassword - ability to change user's password
Self (Self-Membership) - ability to add yourself to a group
```

Pre-requisites

Risks

How to check for

These two queries can be helpful to enumerate attackpaths from any user that is not Domain Admin to become domain admin. These queries have excluded the sessions edge since that is reported in another issue. Remember to change the domain to your domain.

```
MATCH (n:User {admincount: false}),(m:Group {name:"DOMAIN ADMINS@MYDOMAIN.local"}),p=shortestPath((n)-[r:MemberOf|AdminTo|AllExtende

MATCH (n:User {admincount: false}),(m:Group {name:"DOMAIN ADMINS@MYDOMAIN.local"}),p=allshortestPaths((n)-[r:MemberOf|AdminTo|AllExt
```

To find users with DCSync permissions run this. Note that this query is the default query in Bloodhound which is called "Find Principals with DCSync Rights".

```
MATCH (n1)-[:MemberOf|GetChanges*1..]->(u:Domain {name: "MYDOMAIN.local"}) WITH n1,u MATCH (n1)-[:MemberOf|GetChangesAll*1..]->(u) W
```

Find interesting privileges/ACEs that have been configured to DOMAIN USERS group:

```
MATCH (m:Group) WHERE m.name =~ 'DOMAIN USERS@.*' MATCH p=(m)-[r:MemberOf|AdminTo|AllExtendedRights|AddMember|ForceChangePassword|Ge
```

Find interesting edges related to "ACL Abuse" that uprivileged users have against other users:

```
MATCH (n:User {admincount:False}) MATCH (m:User) WHERE NOT m.name = n.name MATCH p=allShortestPaths((n)-[r:AllExtendedRights|ForceCh
```

Find interesting edges related to "ACL Abuse" that unprivileged users have against computers:

```
MATCH (n:User {admincount:False}) MATCH p=allShortestPaths((n)-[r:AllExtendedRights|GenericAll|GenericWrite|Owns|WriteDacl|WriteOwne
```

Find interesting edges related to "ACL Abuse" that unprivileged users have against Groups:

```
MATCH (n:User {admincount:False}) MATCH p=allShortestPaths((n)-[r:AllExtendedRights|GenericAll|GenericWrite|Owns|WriteDacl|WriteOwne
```

Find all Edges that a specific user has against all the nodes (HasSession is not calculated, as it is an edge that comes from computer to user, not from user to computer):

```
MATCH (n:User) WHERE n.name =~ 'HELPDESK@DOMAIN.GR'MATCH (m) WHERE NOT m.name = n.name MATCH p=allShortestPaths((n)-[r:MemberOf|HasS
```

Find if unprivileged users have rights to add members into groups:

```
MATCH (n:User {admincount:False}) MATCH p=allShortestPaths((n)-[r:AddMember*1..]->(m:Group)) RETURN p
```

How to exploit

The commands are all from Powersploits PowerView module.

- ForceChangePassword: The ability to change the target user's password without knowing the current value. Abused with `Set-DomainUserPassword`.
- AddMembers: The ability to add arbitrary users, groups or computers to the target group. Abused with `Add-DomainGroupMember`.
- GenericAll: Full object control, including the ability to add other principals to a group, change a user password without knowing its current value, register an SPN with a user object, etc. Abused with `Set-DomainUserPassword` or `Add-DomainGroupMember`.
- GenericWrite: The ability to update any non-protected target object parameter value. For example, update the "scriptPath" parameter value on a target user object to cause that user to run your specified executable/commands the next time that user logs on. Abused with `Set-DomainObject`.
- WriteOwner: The ability to update the owner of the target object. Once the object owner has been changed to a principal the attacker controls, the attacker may manipulate the object any way they see fit. Abused with Set-DomainObjectOwner.
- WriteDACL: The ability to write a new ACE to the target object's DACL. For example, an attacker may write a new ACE to the target object DACL giving the attacker "full control" of the target object. Abused with `Add-NewADObjectAccessControlEntry`.
- AllExtendedRights: The ability to perform any action associated with extended Active Directory rights against the object. For example, adding principals to a group and force changing a target user's password are both examples of extended rights. Abused with `Set-DomainUserPassword` or `Add-DomainGroupMember`.

Recommendation

References